Temperature Control of High-Performance Multi-core Platforms Using Convex Optimization

Srinivasan Murali[†], Almir Mutapcic[‡], David Atienza[†+],
Rajesh Gupta[§], Stephen Boyd[‡], Luca Benini[¶] and Giovanni De Micheli[†]

[†]LSI, EPFL, Switzerland
[‡]Department of Electrical Engineering, Stanford University, USA
[§]Department of Computer Science and Engineering, UCSD, USA
[+]DACYA, Complutense University of Madrid (UCM), Spain
[¶]DEIS, University of Bologna, Italy

[†]{srinivasan.murali, david.atienza, giovanni.demicheli}@epfl.ch,
[‡]{almirm, boyd}@stanford.edu, [§] rgupta@ucsd.edu, [¶]lbenini@deis.unibo.it

ABSTRACT

With technology advances, the number of cores integrated on a chip and their speed of operation is increasing. This, in turn is leading to a significant increase in chip temperature. Temperature gradients and hot-spots not only affect the performance of the system, but also lead to unreliable circuit operation and affect the life-time of the chip. Meeting the temperature constraints and reducing the hot-spots are critical for achieving reliable and efficient operation of complex multi-core systems. In this work, we present *Pro-Temp*, a convex optimization based method that pro-actively controls the temperature of the cores, while minimizing the power consumption and satisfying application performance constraints. The method guarantees that the temperature of the cores are below a user-defined threshold at all instances of operation, while also reducing the hot-spots. We perform experiments on several realistic multi-core benchmarks, which show that the proposed method guarantees that the cores never exceed the maximum temperature limit, while matching the application performance requirements. We compare this to traditional methods, where we find several temperature violations during the operation of the system.

Keywords

Thermal-aware design, temperature control, dynamic frequency scaling, static and dynamic optimization.

1. INTRODUCTION

With technology scaling, the number of transistors available on a chip and their speed of operation is increasing rapidly. To efficiently utilize the large number of transistors with manageable design complexity and wiring requirements, designers have started integrating multiple processor, memory and hardware cores on the same chip. Today, several commercial multi-core architectures with few cores to several tens of cores are available. Examples include the IBM's Cell [1], Sun's Niagara [2], Tilera's 64-core architecture [4], to name a few.

As the number of cores on the chip and their speed of operation is increasing, the semiconductor industry is facing several technological challenges to build these systems. It is predicted that in the near future, peak power dissipation and consequent thermal implications will be a major performance bottleneck for multi-core systems [5]. Temperature gradients and hot-spots not only affect the performance of the system, but also lead to unreliable circuit operation and affect the life-time of the chip [6]. For ensuring a reliable system operation, the cores need to operate below a maximum temperature value, which is usually between 85°to 110°Celsius [17].

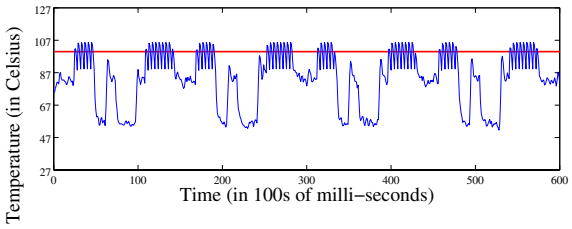

Figure 1: Snap-shot of the thermal behavior for traditional DFS

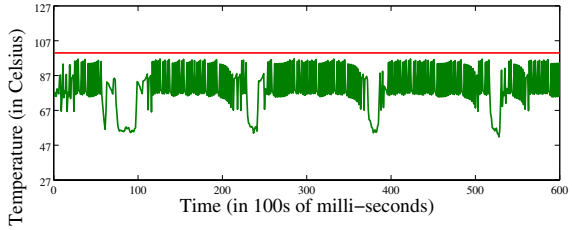

Figure 2: Snap-shot of the thermal behavior for the proposed Pro-Temp method

1.1 Basic Dynamic Frequency Scaling for Thermal Management

Dynamic frequency and voltage scaling (DFS) is a powerful method to reduce the power consumption of the cores, by matching

978-3-9810801-3-1/DATE08 © 2008 EDAA

their performance to application characteristics. In many systems, one or more power management units monitor the application behavior and periodically scale the processor frequencies to meet the required performance level. It is also commonly used to manage the thermal behavior of the cores: when a core reaches a pre-defined temperature threshold level, it is shutdown or its frequency is reduced. However, such a thermal management policy has three major drawbacks:

(1) It is reactive in nature. The cores operate for a long period above the maximum allowable temperature, before the frequency scaling takes place. This is true even when the temperature threshold for frequency scaling is designed to be much lower than the maximum allowable temperature. As an example, in Figure 1, we show the temperature variations on a core utilizing such a scheme (the details of the experiment are presented later, in Section 5). In this example, the maximum allowed temperature is assumed to be 100°Celsius, and frequency scaling is applied when a core reaches 90°Celsius. This example shows that the maximum temperature is violated for sometime, before the DFS forces the core to cool down.

(2) As frequency scaling of the different cores are usually performed independently, the method does not achieve optimal performance for the given temperature constraints. When scaling the frequency of a core, it does not consider the thermal behavior of the other cores.

(3) Finally, the method does not reduce the thermal gradients and hotspots of the chip.

In this work, we present *Pro-Temp*, a convex optimization based method to set the frequencies of the cores. The method pro-actively controls the temperature of the cores, while minimizing the power consumption and satisfying application performance constraints. The proposed method overcomes all the above drawbacks of traditional frequency scaling. It guarantees that the cores always operate below the maximum temperature limit at all time instances of operation and application workloads, while reducing the thermal hotspots and gradients. The frequency assignment for each core also takes into account the global knowledge of the temperature and utilization of all the other cores. In Figure 2, the thermal behavior of the core (from Figure 1) for our Pro-Temp scheme is presented, which shows that the maximum temperature constraint is met at all time instances.

1.2 The Pro-Temp Scheme

The Pro-Temp method consists of two phases: an off-line and an on-line phase. In the off-line phase (done at design time), we determine an optimal frequency assignment for the different processors in order to meet a particular workload constraint, while satisfying the thermal constraints. The frequency assignments are such that, for the entire time-period before the next DFS can be applied, the cores are guaranteed to operate below the maximum temperature value. For this, we use a convex optimization based method [25] to solve the thermal heat flow equations of the chip [24]. In the optimization process, we also reduce the temperature gradients and hot spots on the chip. We apply the method for different workload requirements and starting temperature values of the cores, and store the resulting frequency assignments in a table.

This table is then used at run-time (the on-line phase) by the thermal/power management unit, which periodically applies DFS to the cores. The thermal management unit monitors the application workloads processed by the cores and their temperature values. When DFS is applied, for the current workload and the current maximum temperature value on the chip, the unit chooses the optimal frequency assignment from the table (which was computed in the off-line phase).

To validate our proposed method, we perform experiments on a multi-core model based on the Sun's 8-core Niagara architecture [2]. We perform experiments on several realistic multi-core benchmarks, which show that the proposed method guarantees that the cores never exceed the maximum temperature limit. We compare this to the traditional DFS method, where we find several temperature violations during the operation of the cores. The experiments also show that the methods in a large performance improvement over the traditional DFS mechanism.

Here, we would like to note the fact that there have been numerous works in the field of thermal-aware processor design and task management (please refer Section 2 for details). The existing methods cover a very large design space, from dynamic to static, such as application of DFS [7], task assignment and scheduling policies [8], task migration strategies [10], floorplanning policies [16, 15], etc. For a single system, usually many of these policies are applied together [3]. For example, the DFS method can be applied on a system, where the individual tasks are assigned to processors based on an efficient physical-aware thermal policy, such as presented in [11]. In this work, we only target the design of an efficient pro-active thermal control mechanism that can set the operating frequencies of the cores. This method, can then be used in conjunction with the other thermal policies, such as task migration and task assignment. While our method will guarantee that the temperature of the cores will be less than the maximum value, it is possible to further reduce the temperature gradients on the chip by utilizing other methods in conjunction with ours. Towards this end, in Section 5.4, we show how our method can be integrated with an efficient thermal-aware task assignment strategy.

2. RELATED WORK

A large number of researchers in computer architecture have focused on power management and thermal control for multi-core systems and MPSoCs [3, 11, 21]. While reducing power density has the effect of reducing overall temperature, power-aware design does not directly imply that thermal gradients between different components are minimized or individual hot spots do not appear [17, 3]. Processor power optimizations using frequency and voltage scaling have been proposed in several works [11, 7].

In the last years, research on control policy design for thermal management has received a lot of attention as a collateral effect of increasing power density. In [21, 9], the authors have proposed adaptive mechanisms for thermal management, focusing on handling key micro-architectural hotspots. In addition, task and thread migration techniques have been proposed as basic thermal management schemes in multi-core platforms [10, 3]. They use performance counter-based information or compile-time pre-characterization and achieve significant reductions in localized hotspots. In [22], the authors present a set scheduling mechanisms for MPSoCs to perform temperature management at system-level. In [26], an efficient task assignment policy for multi-core systems is presented.

Finally, several groups have addressed the problem of thermal modeling and simulation at different levels of abstraction [13]-[18]. In [17] a thermal/power model for super-scalar architectures is presented. It predicts temperature variations in processor components and shows effects in leakage power and performance. [18] outlines a simulation model and its validation on embedded cores, which shows temperature variations of 13.6°across the die. Also, [12] models performance and power efficiency in multi-core architectures considering thermal constraints, but it does not propose any optimization policy. At the physical level, various methods have been suggested to model the heat transfer in the sub-

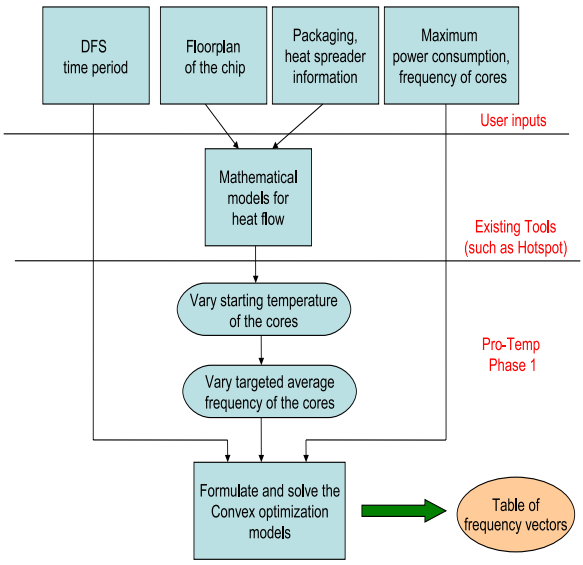

Figure 3: Phase 1 of the Pro-Temp Method

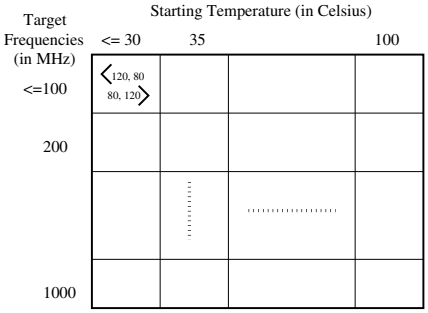

Figure 4: Table structure from the output of Phase 1

strate. Finite-difference time domain [13], finite element [15], and Green-function [16] based algorithms have been applied for on-chip thermal modeling.

3. DESIGN METHODOLOGY

In this section, we describe in detail the operation of the Pro-Temp method.

3.1 System Description and Assumptions

We use the following realistic assumptions for the system characteristics. The system that we target has multiple cores, with each core running a single task or thread. We assume there is at least one thermal sensor for each core, and we utilize a centralized thermal management unit for monitoring the values of the sensors. We assume that one of the processors also acts as a control unit to assign the incoming tasks to the different processors. In our system, we utilize a simple task assignment policy: when a task arrives, the control unit assigns the task to any idle processor. If all the processors are busy, the task is queued up in a task-queue. Please note that other complex task assignment policies can also be used along with our method (an example of which is shown in Section 5.4).

We define the workload of a task as the total amount of time required for running the task, at the highest operating frequency. In most systems, the thermal management and frequency scaling are applied in the milli-seconds scale [3]. We assume that workload of the individual tasks are much smaller than the time window at which the DFS is applied. This is in fact a realistic assumption because in our experiments on multi-core benchmarks, the tasks have a workload of 1 ms - 10 ms.

3.2 Phase 1: Design Time Flow

The off-line phase of the method, which is performed at design time for a multi-core system, is presented in Figure 3. The floorplan of the chip and the maximum power consumption and maximum operating frequencies of the cores are obtained as inputs. Based on the packaging and the heat spreader used in the system, the thermal models that can track the temperature variations of the cores are obtained. For this, we use existing tools and methods, such as the Hotspot [17] and the MPSoC thermal modeling tool presented in [19]. The time period at which the DFS needs to be applied is also obtained as an input.

The convex optimization procedure is solved for different starting temperature values of the cores. As repeating the procedure for all possible combinations of the starting temperatures of the different cores leads to exponential complexity and infeasibility, we simplify the process by only iterating on one temperature value. During run time, this translates to the maximum temperature value across all the cores. As the optimization method also minimizes the temperature gradient across the cores, from our experiments we found that this simplification has very little impact on the quality of the results (see Section 5 for more details).

The workload requirement of the tasks in a time period directly translates to a frequency requirement on the processors. As an example, assume 200 tasks are assigned to 4 processors in a DFS time window of 100 ms, with each task having a workload of 1 ms at 1 GHz frequency. The average speed of the processors should be 500 MHz to satisfy the workload characteristics of the tasks. In our design flow, we vary the required average frequency of the processors and apply the convex optimization for each design point. If the required frequency point cannot be supported, the optimization notifies an infeasible solution. The mathematical models of the convex optimization problem are explained in detail in Section 4.

The output of Phase 1 is a table of frequency vectors, with a different frequency vector for each starting temperature value and required average frequency, as shown in Figure 4.

3.3 Phase 2: Run Time Control

In the on-line phase of our method, the thermal management unit utilizes the table obtained in Phase 1 to set the frequencies of the processors. The DFS is applied periodically, at a pre-defined time period. In each time period, the utilization of the different processors is tracked by the thermal management unit. The unit also monitors the workload of the tasks waiting in the task queue, to be executed by the cores. Based on these information, the required average operating frequency across all the processors for the next period is calculated by the unit. Before applying the DFS, the unit gathers the temperature information of the processors and finds the maximum temperature across the cores. Based on this temperature value and the required average frequency of the cores, the unit chooses the frequency assignment for the processors from the table. If the frequency point cannot be supported, the unit chooses the next lower frequency point in the table that can support the temperature constraints.

4. CONVEX MODELS

In this section, we first briefly describe the convex models. A more detailed description of the model is presented in [24]. We

then show the additional constraints that are added to achieve uniform thermal gradient across the cores. In many existing multi-core architectures, such as [1] or [2], in order to simplify the design, the operating frequencies of all the cores are the same. For such systems, when DFS is applied, the core frequencies are varied uniformly. We also show how such uniform frequency setting can be achieved in the convex model.

Let n be the number of cores in the design. We denote the power consumption of a core i as p_i. The set of cores that are adjacent to a core i is represented by Adj_i. The temperature of core i at time $k+1$ is given by the thermal equation:

$$t_{k+1,i} = t_{k,i} + \sum_{\forall j \in\ Adj_i} a_{i,j}(t_{k,j} - t_{k,i}) \ + \ b_i p_i \qquad (1)$$

In this formulation the temperature of core i at the current time instant depends on the temperature of itself and its neighboring cores in the previous time instant, as well as on the power consumption of the core. The proportionality constants $a_{i,j}$ and b_i are based on the thermal behavior of the chip, and are calculated as presented in [17], [19].

The total number of time-steps used for the thermal calculations depends on the time-period at which DFS is applied. In our experiments, in order to achieve numerical stability, the thermal equation (Equation 1) had to be solved with a time step of 0.4 ms. If the DFS scheme is applied every 100 ms, then the total number of time-steps needed is 250. We denote the number of time-steps needed by m, and is obtained as an input to the optimization procedure.

The initial operating temperature of the cores, $t_{0,i}$, is set to t_{start}, which is an input to the procedure (please refer to Figure 3). The required target operating frequency of the cores (denoted by f_{target}) and the maximum allowable temperature (denoted by t_{max}) are also obtained as inputs.

The frequency of operation of core i is represented by the variable f_i, $i = 1, \ldots, n$. The objective of the optimization procedure is to find the optimal f_i values, such that the average of the frequencies of the cores meets the targeted frequency (f_{target}), while minimizing the power consumption and satisfying temperature constraints.

The operating voltage of a processor depends on the operating frequency, and this dependence varies with different process and technology generations. In this work we assume that the square of the voltage scales linearly with the frequency of operation, as it is a common method to scale voltage [23]. The power consumption values at different time-instances can be obtained by quadratically scaling the power consumption of the processors at $f_{\max}$ (which is denoted by p_{max}), *i.e.*,

$$p_i = p_{\max}\ f_i^2 / f_{max}^2, \forall i \qquad (2)$$

The convex model to solve the thermal and workload constrained, power optimization problem is presented below:

$$\begin{array}{ll} \text{min:} & \sum_{i=1}^{n} p_i \\ \text{s.t.} & t_{0,i} \ = \ t_{start}, \forall i \\ & t_{k+1,i} = t_{k,i} + \sum_{\forall j \in Adj_i} a_{i,j}(t_{k,j} - t_{k,i}) + b_i p_i \forall i,\ k \\ & t_{k,i} \leq t_{\max}, \forall k,\ i \\ & p_{\max}\ f_i^2 / f_{\max}^2 <= p_{i,k}, \forall i,\ k \\ & \sum_{i=1}^{n} f_i \ \geq \ n \times f_{target} \\ & f_i \ \geq \ 0,\ \forall i \end{array} \qquad (3)$$

In order to achieve uniform spatial temperature gradients across the cores, the following additional equations are added to the model:

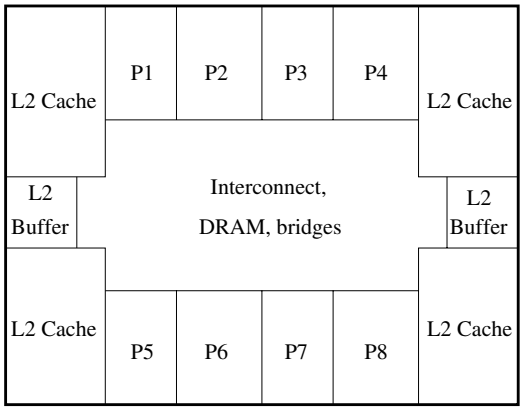

Figure 5: The floorplan of the Sun's Niagara multi-core architecture. The processing cores are represented by P1-P8

$$t_{k,i} - t_{k,j} \ \leq \ tgrad, \forall i, j \in 1 \cdots n, \forall k \qquad (4)$$

and the objective function is changed to minimize the gradient as well:

$$\text{min:}(\sum_{i=1}^{n} p_i \ + \ tgrad) \qquad (5)$$

5. EXPERIMENTAL RESULTS

For the experiments, we consider the 8-core Niagara architecture from Sun Microsystems [2]. The floorplan of the architecture is presented in Figure 5. The architecture has different versions, with the processors supporting a maximum operating frequency between 1 GHz to 1.4 GHz. In this work, we assume the maximum frequency of the processors to be 1 GHz and the maximum power consumption of each processor core at this frequency to be 4 W. The power consumption of the other cores on the system is around 30% of the power consumption of the processing cores [2].

We use the execution characteristics of tasks from a mix of different benchmarks, ranging from web-accessing to playing multimedia files [26]. The maximum task/thread lengths of the benchmarks is around 10 ms. The experiments are conducted using a large trace with around 60,000 tasks, modeling several hundred seconds of actual system execution.

We implemented a simulator to model the task assignment and execution on the different cores. For simulating the temperature profiles of the cores, we use the thermal models presented in [19]. We also verified our simulator using the thermal models from the Hotspot simulator [17].

5.1 Design Time

The convex models presented in the previous section can be solved with polynomial (in the number of variables and constraints) time complexity using interior point methods [25]. To solve the models, we use CVX [27], an efficient convex optimization solver. For our experimental set-up, the solver takes less than 2 minutes to determine the optimal solution. As the optimization models are solved for each temperature and frequency point (as presented earlier in Figure 3), the total time taken to perform phase 1 of the method is few hours. Please note that phase 1 is performed only once for a system at design time and the timing overhead for this is negligible.

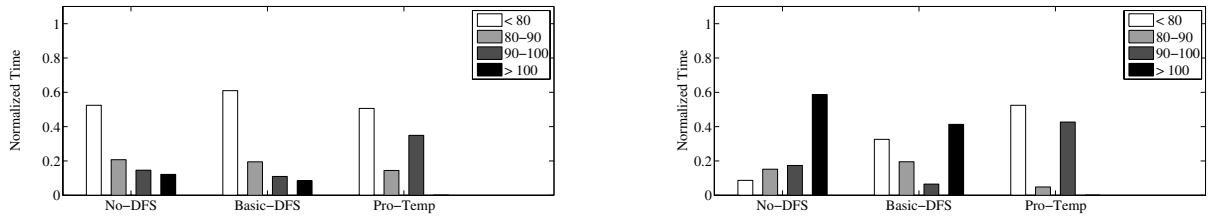

Figure 6: The percentage time spent on average by the cores at different temperature points: (a) for a mix of tasks from different benchmarks, and (b) for the most computation intensive benchmark

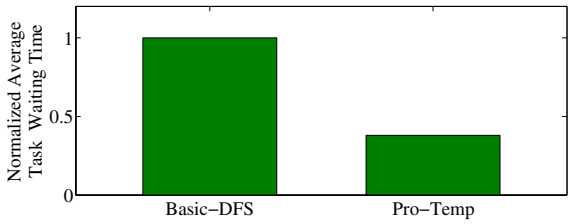

Figure 7: Performance comparisons with the Basic-DFS method

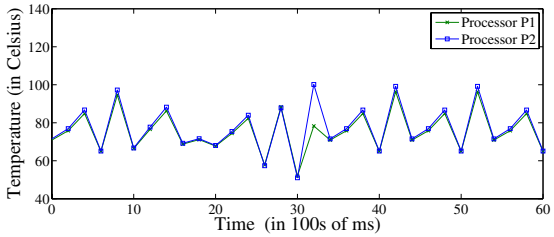

Figure 8: The temperatures of processors P1 and P2 over time.

5.2 Comparisons with Existing Methods

We implemented our proposed Pro-Temp scheme in the simulator. For comparison purposes, we also implemented a traditional DFS scheme (referred to as the *Basic-DFS* scheme), where the frequencies of the cores are matched to the application performance levels. The temperature control performed when a core reaches a threshold temperature level. In this case, the core js shuts down for the time-period until the next DFS is applied. The maximum temperature constraint on the cores is set to 100°Celsius. We assume that the temperature threshold level for application of traditional DFS to be 90°Celsius. The snap-shots of the temperature of one of the processors (processor P1) for the traditional DFS and the proposed Pro-Temp scheme were presented earlier, in Figures 1 and 2.

In Figure 6, we plot the percentage time the processors (averaged across all the processors) spent at different temperature ranges. For reference, we also plot the values when no temperature control is applied (referred to as the *No-TC* method). In this scheme, the frequencies are scaled only to match the application characteristics.

As seen from the figure, the Pro-Temp method always ensures that the processors are below the maximum temperature of 100°Celsius, while the No-TC and Basic-DFS spend a significant amount of time above the maximum temperature. For the most computation intensive benchmark, the Basic-DFS scheme spends

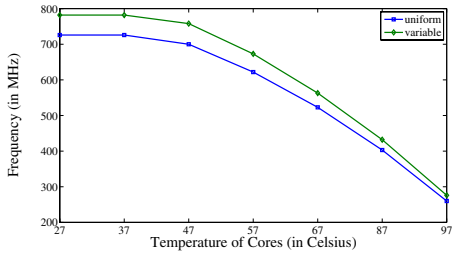

Figure 9: The average frequency of the 8 processors, when uniform and variable frequency assignments are applied.

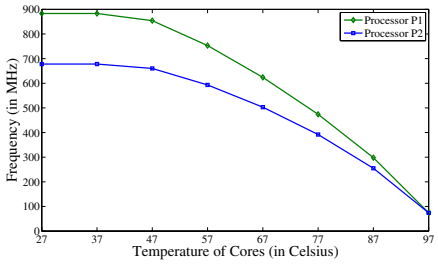

Figure 10: The operating frequency of the processors 1 and 2 as computed by our method

up to 40% of the time above the maximum threshold.

The performance of the system is also much higher for the Pro-Temp scheme. In Figure 7, we plot the average waiting times of the tasks for the scheme, normalized with respect to the Basic-DFS technique. The proposed scheme results in 60% reduction in the task waiting times. This is because, with the Basic-DFS scheme, the cores operate fast until they reach the maximum threshold limit. After that, they are shutdown until they cool down. As the cooling period is relatively longer than the period in which they heat up, for computation intensive workloads, the Basic-DFS has a poorer performance when compared to the Pro-Temp scheme.

The temperature values of two of the processors over time, for the Pro-Temp method is shown in Figure 8. From the figure, we can see that the temperature gradient across the processors is low.

5.3 Uniform Vs Variable Frequency Setting

When DFS is applied, the frequencies of all the processors could either be set to the same value or they can be varied. From the floorplan of the system presented in Figure 5, we find that the processors P1, P4, P5 and P8 are near cooler caches, while the other four processors are sandwiched by processing cores on two sides.

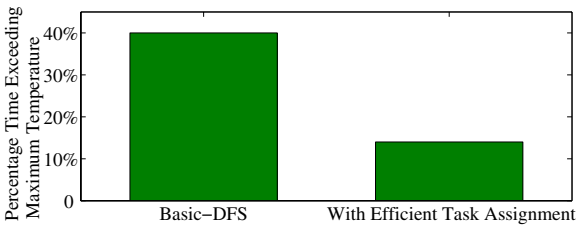

Figure 11: Effect of efficient task assignment

The processors near the cooler caches can more easily dissipate their heat than the other processors. To compensate for the thermal imbalance, the cores in the middle need to operate at a lower frequency than the cores at the periphery. We modeled both the uniform and the variable frequency assignment policies and performed experiments using the benchmarks. For a given starting temperature and maximum temperature constraint, a non-uniform frequency assignment can support a higher average workload than the uniform assignment. This is shown in Figure 9.

The frequency assignment for processors P1 and P2, obtained by our convex optimization procedure for the non-uniform frequency assignment scheme is presented in Figure 10. From the figure, we can see that the processor P1 runs significantly faster than P2 to achieve a similar thermal behavior.

5.4 Effect of Assignment Policy

An efficient thermal policy for assignment of tasks on to cores is presented in [26]. We integrated this assignment with the Basic-DFS and our Pro-Temp methods. When the assignment policy is applied, the percentage of time the Basic-DFS is above the maximum temperature reduces, as shown in Figure 11 (for the high workload benchmark). However, due to the burstiness in the task arrival pattern, still the method results in cores spending a significant time over the maximum temperature. As noted earlier, the Pro-Temp method always results in the chip operating below the maximum temperature at all instances. However, with the integration of the efficient task assignment policy with our Pro-Temp method, the spatial temperature difference across the cores reduces further (by 16%).

6. CONCLUSIONS

Temperature control of multi-core architectures is critical for achieving a reliable and high performance operation. In this paper, we have presented a convex optimization based method to proactively control the frequencies of the cores, such that the temperature constraints are met at all time instances of operation. Our novel approach solves the thermal control problem in two phases: in the first phase, at design time, the set of feasible frequencies for the cores for different temperature and workload constraints are obtained by solving convex optimization models for the problem. In the next phase, at run time, the frequency values from the previous phase are used to match the current system workload and operating conditions. Our experiments on realistic benchmarks show that optimal temperature control is achieved by our method, while ensuring high performance operation of the system.

7. ACKNOWLEDGMENTS

This research is supported by a grant from the Fonds National Suisse (FNS, Grant 20021-109450/1).

8. REFERENCES

[1] D. Pham, et al, “Design and Implementation of a First-Generation Cell Processor”, Proc. IEEE ISSCC, 2005.
[2] P. Kongetira, K. Aingaran, and K. Olukotun, “Niagara: A 32-way multithreaded SPARC processor”, IEEE Micro, March-April 2005.
[3] J. Donald, and M. Martonosi, “Techniques for multi-core thermal management: Classification and new exploration”, Proc. of ISCA, 2006.
[4] Tilera Corporation, “Tilera’s 64-core architecture”, www.tilera.com/products/processors.php, 2007.
[5] S. Borkar, “Design challenges of technology scaling”, IEEE Micro, July-Aug 1999.
[6] O. Semenov et al., “Impact of self-heating effect on long-term reliability and performance degradation in CMOS circuits”, IEEE Transactions on Devices and Materials, March 2006.
[7] C J. Hughes, J. Srinivasan, and S. V. Adve, “Saving energy with architectural and frequency adaptations for multimedia applications, Proc MICRO, 2001.
[8] Y. Xie, and W.-L. Hung, “Temperature-Aware Task Allocation and Scheduling for Embedded MPSoC Design”, Kluwer J. VLSI Signal Process. Syst., 2006.
[9] F. Bellosa, A. Weissel, M. Waitz, and S. Kellner. “Event-driven energy accounting for dynamic thermal management”, Proc. of COLP, 2003.
[10] P. Chaparro, J. Gonzalez, G. Magklis, Q. Cai, and A. Gonzalez. “Understanding the thermal implications of multi-core architectures”. IEEE TPDS, 2007.
[11] R. Mukherjee, and S. O. Memik, “Physical aware frequency selection for dynamic thermal management in multi-core systems”, Proc. of ICCAD, 2006.
[12] J. Li and J. Martinez, “Power-performance implications of thread-level parallelism in chip multiprocessors”, Proc. ISPASS, 2005.
[13] T.-Y. Wang and C.-P. Chen, “3-d thermal-adi: A linear-time chip level transient thermal simulator,” IEEE TCAD, December 2002.
[14] J. Deeney, “Thermal modeling and measurement of large high power silicon devices with asymmetric power distribution,” Proc. of the International Symposium on Microelectronics, 2002.
[15] B. Goplen and S. Sapatnekar, “Efficient thermal placement of standard cells in 3d ics using a force directed approach”, Proc. ICCAD 2003.
[16] Y. Zhan and S. Sapatnekar, “Fast computation of the temperature distribution in vlsi chips using the discrete cosine transform and table look-up”, ASPDAC 2005.
[17] K. Skadron et al., “Temperature-aware microarchitecture: Modeling and implementation”, TACO, 2004.
[18] H. Su, et al. “Full chip leakage estimation considering power supply and temperature variations”, Proc. of ISLPED, 2003.
[19] G. Paci, et al, “Exploring temperature-aware design in low-power MPSoCs”, Proc. of DATE, 2006.
[20] J. Srinivasan and S. V. Adve, “Predictive dynamic thermal management for multimedia applications”, ICS03, June 2003.
[21] D. Brooks and M. Martonosi, “Dynamic thermal management for high-performance microprocessors,” HPCA, 2001.
[22] W. Hung et al., “Thermal-aware allocation and scheduling for systems-on-chip”, DATE, 2005.
[23] S. Murali, et al., “Mapping and configuration methods for multi-use-case networks on chips”, Proc. of ASP-DAC, 2006.
[24] S. Murali et al., “Temperature-aware processor frequency assignment for MPSoCs using convex optimization”, Proc. CODES-ISSS, pp 111-116, 2007.
[25] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
[26] A. K. Coskun, T. Simunic Rosing, and K. Whisnant, “Temperature Aware Task Scheduling in MPSoCs”, Proc. of DATE, 2007.
[27] M. Grant, S. Boyd, and Y. Ye. CVX: Matlab software for disciplined convex programming, version 1.0 beta 3. Available at www.stanford.edu/~boyd/cvx/.